select FirstName,LastName,City,State

from Person p

left join Address a

on p.PersonId=a.Personid

select(

select distinct Salary

from Employee

order by Salary desc limit 1 offset 1) as SecondHighestSalary

CREATE FUNCTION getNthHighestSalary(N INT) RETURNS INT

BEGIN

    SET N=N-1;

    RETURN (

        select distinct Salary From Employee order by Salary desc limit 1 offset N

  );

END

select Score, dense\_rank()

over(order by Score desc) as 'Rank'

from Scores

select distinct l1.Num as ConsecutiveNums

from Logs l1

join Logs l2 on l1.Num=l2.Num and l2.id=l1.id+1

join Logs l3 on l2.Num=l3.Num and l3.id=l2.id+1

select e1.Name as Employee

from Employee e1

join Employee e2

on e1.ManagerId=e2.Id

where e1.Salary>e2.Salary

select Email

from Person

group by Email

having count(\*)>1

select Name as Customers

from Customers

where id not in (select CustomerId from Orders)

select d.Name as Department,e.Name as Employee, Salary

from Employee e

join Department d

on e.DepartmentId=d.Id

where (DepartmentId,Salary) in

(select DepartmentId,max(Salary)

from Employee e

group by DepartmentId)

select d.Name as Department,Employee,Salary

from(

select DepartmentId,Name as Employee,Salary,

    dense\_rank() over(partition by DepartmentId order by Salary desc) as rnk

from Employee) t

join Department d on t.DepartmentId=d.Id

where rnk <=3

order by d.Id, Salary desc

delete p1

from Person p1

join Person p2

where p1.Email=p2.Email and p1.id>p2.id

select w2.Id

from Weather w1

join Weather w2

on datediff(w2.RecordDate,w1.RecordDate)=1

where w2.Temperature>w1.Temperature

select Request\_at as day, round(sum(case when Status like "%cancelled%" then 1 else 0 end)/count(\*),2) as "Cancellation Rate"

from Trips

where Client\_id not in (select Users\_id from Users where Banned="Yes") and Request\_at between '2013-10-01' and '2013-10-03'

group by Request\_at

select player\_id,min(event\_date) as first\_login

from Activity

group by player\_id

select player\_id,device\_id

from Activity

where (player\_id,event\_date) in (select player\_id, min(event\_date) from Activity group by player\_id)

select player\_id, event\_date,

sum(games\_played) over(partition by player\_id order by event\_date) as games\_played\_so\_far

from Activity

select round(count(\*)/(select count(distinct player\_id) from Activity),2) as fraction

from Activity a1

join Activity a2 on a1.player\_id=a2.player\_id and datediff(a2.event\_date,a1.event\_date)=1

where (a1.player\_id, a1.event\_date) in (select player\_id,min(event\_date) from Activity group by player\_id)

select Id,Company,Salary from

(

select Id,Company,Salary,

row\_number() over(partition by Company order by Salary) as rnk,

count(Salary) over(partition by Company) as cnt from Employee

) t

where rnk in (cnt/2,cnt/2+1,cnt/2+0.5)

select e1.Name

from Employee e1

join Employee e2

on e1.Id=e2.ManagerId

group by e1.Id

having count(e2.Id) >=5

**571**

select avg(number) as median

from(

select Number, frequency,

        sum(frequency) over(order by number asc) as t1,

        sum(frequency) over(order by number desc) as t2

from Numbers

order by Number) t

where t1>=(select sum(frequency) from Numbers)/2 and t2>=(select sum(frequency) from Numbers)/2

select Name

from Candidate

where id = (select CandidateId from Vote group by CandidateId order by count(id) desc limit 1)

select e.name,bonus

from Employee e

left join Bonus b on e.empId=b.empId

where bonus <1000 or bonus is null

select question\_id as survey\_log

from (

select question\_id, sum(case when action="answer" then 1 else 0 end)/count(\*) as rate

from survey\_log

group by question\_id) t

order by rate desc limit 1

**579**

select Id, Month,

        sum(Salary) over(partition by Id order by Month rows 2 preceding) Salary

from Employee

where (Id, Month) not in

(

    select Id, max(Month)

    from Employee

    group by Id

)

order by Id, Month desc

select dept\_name, count(student\_id) as student\_number

from department d

left join student s on d.dept\_id=s.dept\_id

group by dept\_name

order by count(student\_id) desc, dept\_name

select name

from customer

where referee\_id <> 2 or referee\_id is null

**585**

select sum(TIV\_2016) as TIV\_2016

from insurance

where TIV\_2015 in (select TIV\_2015 from insurance group by TIV\_2015 having count(\*)>1)

and concat(lat,lon) in (select concat(lat,lon) from insurance group by concat(lat,lon) having count(\*)=1)

select customer\_number

from orders

group by customer\_number

order by count(order\_number) desc limit 1

select name,population,area

from world

where population>25000000 or area>3000000

select class

from courses

group by class

having count(distinct student) >= 5

select round(

    ifnull(

    (select count(distinct requester\_id ,accepter\_id) from request\_accepted) /

    (select count(distinct sender\_id ,send\_to\_id) from friend\_request)

    ,0)

,2) as accept\_rate

select id,visit\_date,people

from stadium

where id in (select s1.id as id

from stadium s1

join stadium s2 on s2.id=s1.id+1

join stadium s3 on s3.id=s2.id+1

where s1.people>=100 and s2.people>=100 and s3.people>=100

union

select s2.id

from stadium s1

join stadium s2 on s2.id=s1.id+1

join stadium s3 on s3.id=s2.id+1

where s1.people>=100 and s2.people>=100 and s3.people>=100

union

select s3.id

from stadium s1

join stadium s2 on s2.id=s1.id+1

join stadium s3 on s3.id=s2.id+1

where s1.people>=100 and s2.people>=100 and s3.people>=100)

select requester\_id as id, count(accepter\_id) as num

from(

select requester\_id,accepter\_id from request\_accepted

union all

select accepter\_id,requester\_id from request\_accepted) t

group by requester\_id

order by count(accepter\_id) desc limit 1

select distinct c1.seat\_id

from cinema c1

join cinema c2

on c1.free=1 and c2.free=1 and (c1.seat\_id=c2.seat\_id+1 or c1.seat\_id=c2.seat\_id-1)

select name

from salesperson

where sales\_id not in (select sales\_id from orders o join company c on o.com\_id=c.com\_id where c.name="RED")

select id,if(isnull(p\_id),"Root",if(id in (select p\_id from tree),"Inner","Leaf")) as Type

from tree

select x,y,z,if(x+y>z and x+z>y and y+z>x,"Yes","No") as triangle

from triangle

select distinct f2.followee as follower,count(distinct f2.follower) as num

from follow f1

join follow f2 on f1.follower=f2.followee

group by f2.followee

select t1.pay\_month,t1.department\_id,

case

when dep\_salary > comp\_salary then "higher"

when dep\_salary<comp\_salary then "lower"

else "same" end as comparison

from(

select department\_id,date\_format(pay\_date,"%Y-%m") as pay\_month, avg(amount) as dep\_salary

from salary s

join employee e on s.employee\_id=e.employee\_id

group by date\_format(pay\_date,"%Y-%m"),department\_id) t1

join(

select date\_format(pay\_date,"%Y-%m") as pay\_month, avg(amount) as comp\_salary

from salary

group by date\_format(pay\_date,"%Y-%m")

) t2

on t1.pay\_month=t2.pay\_month

**618**

select

max(case when continent = 'America' then name else null end) as America,

max(case when continent = 'Asia' then name else null end)as Asia,

max(case when continent = 'Europe' then name else null end) as Europe

from

(select \*, row\_number() over(partition by continent order by name) rk

from student) t

group by rk

select max(num) as num

from my\_numbers

where num in (select num from my\_numbers group by num having count(\*)=1)

select \*

from cinema

where description <> "boring" and id%2<>0 order by rating desc

**626**

SELECT (CASE

            WHEN id%2=1 AND id=(SELECT COUNT(\*) FROM seat) THEN id

            WHEN id%2=1 THEN id+1

            ElSE id-1

        END) AS id, student

FROM seats

ORDER BY id

update salary

set sex=if(sex="m","f","m")

select customer\_id

from Customer

group by customer\_id

having count(distinct product\_key) = (select count(distinct product\_key) from product)

select actor\_id,director\_id

from ActorDirector

group by concat(actor\_id,'-',director\_id)

having count(timestamp) >=3

select product\_name,year,price

from sales s

join product p on s.product\_id=p.product\_id

select product\_id,sum(quantity) as total\_quantity

from sales

group by product\_id

select product\_id,year as first\_year,quantity,price

from Sales

where (product\_id,year) in (select product\_id,min(year) from Sales group by product\_id)

select project\_id,round(avg(experience\_years),2) as average\_years

from Project p

join Employee e on p.employee\_id=e.employee\_id

group by project\_id

select project\_id

from Project p

group by project\_id

having count(\*) = (select count(\*) from Project group by project\_id order by count(\*) desc limit 1)

select p.project\_id,p.employee\_id

from project p

join employee e on e.employee\_id=p.employee\_id

where (p.project\_id,e.experience\_years) in

(

select project\_id,max(experience\_years)

from employee e

join project p on e.employee\_id=p.employee\_id

group by project\_id)

OR

select project\_id,e.employee\_id,

dense\_rank() over(partition by project\_id order by experience\_years desc) as rnk

from project p

join employee e on p.employee\_id=e.employee\_id

where rnk=1

select seller\_id

from sales

group by seller\_id

having sum(price) = (select sum(price) from sales group by seller\_id order by sum(price) desc limit 1)

select buyer\_id

from sales s

join product p on s.product\_id=p.product\_id

group by buyer\_id

having sum(case when product\_name="S8" then 1 else 0 end)>0 and sum(case when product\_name="iPhone" then 1 else 0 end)=0

select product\_id,product\_name

from Product

where product\_id in (

select product\_id

from Sales

group by product\_id

having min(sale\_date) >= "2019-01-01" and max(sale\_date) <= "2019-03-31" )

select first\_date as install\_dt, count(\*) as installs,

round(count(a.event\_date)/count(\*),2) as Day1\_retention

from (

select player\_id,min(event\_date) as first\_date

from Activity

group by player\_id) t

left join Activity a

on datediff(a.event\_date,t.first\_date)=1 and a.player\_id=t.player\_id

group by first\_date

select b.book\_id,b.name

from (

select book\_id,name

from Books

where available\_from <='2019-05-23') b

left join (

select book\_id,sum(quantity) as total

from orders

where dispatch\_date between '2018-06-23' and '2019-06-23'

group by book\_id) o

on b.book\_id=o.book\_id

where ifnull(total,0)<10

select login\_date,count(user\_id) as user\_count

from(

select distinct user\_id,min(activity\_date) as login\_date

from traffic

where activity="login"

group by user\_id

having datediff('2019-06-30',min(activity\_date)) <=90) t

group by login\_date

order by login\_date

select student\_id,course\_id,grade

from (

select student\_id, course\_id,grade, row\_number() over(partition by student\_id order by grade desc,course\_id) as rnk

from enrollments) t

where rnk=1

select extra as report\_reason, count(distinct post\_id) as report\_count

from actions

where datediff('2019-07-05',action\_date)=1 and action="report" and extra is not null

group by extra

select business\_id

from(

select business\_id,e.event\_type,occurences,avg\_time,

sum(case when occurences>avg\_time then 1 else 0 end) as times

from events e

join (select event\_type,avg(occurences) as avg\_time from events group by event\_type) t

on e.event\_type=t.event\_type

group by business\_id) t1

where times>=2

**1127**

select round(avg(rate)\*100,2) as average\_daily\_percent

from (select action\_date, (count(distinct r.post\_id)/count(distinct a.post\_id)) as rate

from Actions a

left join Removals r

on a.post\_id=r.post\_id

where action="report" and extra="spam"

group by action\_date) t

select activity\_date as day, count(distinct user\_id) as active\_users

from Activity

where datediff('2019-07-27',activity\_date)<30

group by activity\_date

select ifnull(round(sum(sessions)/count(user\_id),2),0) as average\_sessions\_per\_user

from (select distinct user\_id, count(distinct session\_id) as sessions

from Activity

where datediff('2019-07-27',activity\_date)<30

group by user\_id) t

select distinct author\_id as id

from Views

where author\_id=viewer\_id

order by id

select distinct viewer\_id as id

from Views

group by viewer\_id,view\_date

having count(distinct article\_id)>=2

order by id

select u.user\_id as buyer\_id,u.join\_date,count(o.order\_id) as orders\_in\_2019

from users u

left join (

    select \* from Orders where year(order\_date)='2019') o

on u.user\_id=o.buyer\_id

group by u.user\_id

select u.user\_id as seller\_id,

case when u.favorite\_brand=i.item\_brand then "yes" else "no" end as 2nd\_item\_fav\_brand

from users u

left join(

select seller\_id,item\_id,row\_number() over(partition by seller\_id order by order\_date) ods

from orders) t

on u.user\_id=t.seller\_id and t.ods=2

left join items i on i.item\_id=t.item\_id

**1164**

select distinct p.product\_id,

case when t.lastest\_date is null then 10 else p.new\_price end as price

from Products p

left join(

select product\_id, max(change\_date) as lastest\_date

from products

where change\_date <= '2019-08-16'

group by product\_id) t

on p.product\_id=t.product\_id

where t.lastest\_date=p.change\_date or t.lastest\_date is null

select round((count(\*)/(select count(\*) from Delivery))\*100,2) as immediate\_percentage

from Delivery

where order\_date=customer\_pref\_delivery\_date

select round(100\*count(\*)/(select count(distinct customer\_id) from Delivery),2) as immediate\_percentage

from Delivery d1

left join (select customer\_id, min(order\_date) as first\_order

from Delivery

group by customer\_id) d2

on d1.customer\_id=d2.customer\_id

where d1.order\_date=d2.first\_order and d1.order\_date=d1.customer\_pref\_delivery\_date

select id,

    sum(case when month="Jan" then revenue else null end) as Jan\_Revenue,

    sum(case when month="Feb" then revenue else null end) as Feb\_Revenue,

    sum(case when month="Mar" then revenue else null end) as Mar\_Revenue,

    sum(case when month="Apr" then revenue else null end) as Apr\_Revenue,

    sum(case when month="May" then revenue else null end) as May\_Revenue,

    sum(case when month="Jun" then revenue else null end) as Jun\_Revenue,

    sum(case when month="Jul" then revenue else null end) as Jul\_Revenue,

    sum(case when month="Aug" then revenue else null end) as Aug\_Revenue,

    sum(case when month="Sep" then revenue else null end) as Sep\_Revenue,

    sum(case when month="Oct" then revenue else null end) as Oct\_Revenue,

    sum(case when month="Nov" then revenue else null end) as Nov\_Revenue,

    sum(case when month="Dec" then revenue else null end) as Dec\_Revenue

from Department

group by id

select date\_format(trans\_date,"%Y-%m") as month, country, count(\*) as trans\_count, sum(state="approved") as approved\_count, sum(amount) as trans\_total\_amount, sum(case when state="approved" then amount else 0 end) as approved\_total\_amount

from transactions

group by country, date\_format(trans\_date,"%Y-%m")

select group\_id,player\_id

from (

select p.group\_id,p.player\_id,rank() over(partition by group\_id order by total\_score desc,p.player\_id) as rnk

from players p

left join (

select player\_id, sum(score) as total\_score

from(

select first\_player as player\_id,first\_score as score from matches

union all

select second\_player as player\_id,second\_score as score from matches) t1

group by player\_id) t2 on t2.player\_id=p.player\_id) t3

where rnk=1

select person\_name

from(

select person\_name,sum(weight) over(order by turn) as sum\_weight

from queue) t

where sum\_weight<=1000

order by sum\_weight desc limit 1

select date\_format(trans\_date,"%Y-%m") as month, country, sum(state="approved") as approved\_count,sum(case when state="approved" then amount else 0 end) as approved\_amount, sum(state="cb") as chargeback\_count, sum(case when state="cb" then amount else 0 end) as chargeback\_amount

from(

select \* from Transactions

union all

select trans\_id,country,"cb" as state,amount,c.trans\_date

from Chargebacks c

left join transactions t

on c.trans\_id=t.id) t1

group by country, month

having approved\_amount or chargeback\_amount

select query\_name, round(sum(rating/position)/count(\*),2) as quality, round(100\*sum(rating<3)/count(\*),2) as poor\_query\_percentage

from Queries

group by query\_name

select team\_id, t.team\_name,

ifnull(sum(case when t1.host\_goals>t1.guest\_goals then 3

    when t1.host\_goals=t1.guest\_goals then 1

    else 0 end),0) as num\_points

from Teams t

left join

(select host\_team, guest\_team, host\_goals, guest\_goals from Matches

union all

select guest\_team, host\_team, guest\_goals, host\_goals from Matches) t1

on t.team\_id=t1.host\_team

group by t.team\_id

order by num\_points desc, t.team\_id

select p.parent\_id as post\_id,count(distinct s.sub\_id) as number\_of\_comments

from(

select distinct sub\_id as parent\_id from submissions where parent\_id is null) p

left join

(select distinct sub\_id,parent\_id

from submissions

where parent\_id is not null) s on p.parent\_id=s.parent\_id

group by p.parent\_id

select t.product\_id,round(sum(price\*units)/sum(units),2) as average\_price

from(

select p.product\_id, price,units

from Prices p

join UnitsSold u

on p.product\_id=u.product\_id and (u.purchase\_date between p.start\_date and p.end\_date)

) t

group by t.product\_id

select distinct page\_id as recommended\_page

from likes

where user\_id in (

select distinct user2\_id

from(

select user1\_id,user2\_id from friendship

union all

select user2\_id,user1\_id from friendship) f

where user1\_id=1) and page\_id not in (select page\_id from likes where user\_id=1)

select employee\_id

from Employees

where employee\_id<>1 and (manager\_id=1 or manager\_id in (select employee\_id from Employees where manager\_id=1) or manager\_id in (select employee\_id from Employees where manager\_id=1 or manager\_id in(select employee\_id from Employees where manager\_id=1)))

select s.student\_id, s.student\_name,s.subject\_name,ifnull(count(e.subject\_name),0) as attended\_exams

from

(select \*

from Students, Subjects) s

left join Examinations e

on s.student\_id=e.student\_id and s.subject\_name=e.subject\_name

group by s.student\_name,s.subject\_name

order by s.student\_id,s.subject\_name

**1285 row\_number(), over(order\_by xxx partition by xxx)**

select c.country\_name,

(case when avg(weather\_state)<=15 then "Cold"

    when avg(weather\_state)>=25 then "Hot"

    else "Warm" end) as weather\_type

from Countries c

left join Weather w

on c.country\_id=w.country\_id

where w.day between '2019-11-01' and '2019-11-30'

group by c.country\_id

select employee\_id,team\_size

from Employee e

join

(select team\_id,count(\*) as team\_size

from Employee

group by team\_id) t

on t.team\_id=e.team\_id

select gender,day,sum(score\_points)

 over(partition by gender order by day) as total

from Scores

**算running total用 sum() over(partition by xxx order by xxx)最快**

**1321**

select distinct visited\_on, sum\_amount as amount, round(average\_amount,2) as average\_amount

from

(select visited\_on,

**sum(amount) over(order by visited\_on rows 6 preceding) as sum\_amount**,

    avg(amount) over(order by visited\_on rows 6 preceding) as average\_amount

from(

select visited\_on,sum(amount) as amount

from Customer

group by visited\_on) t)t2

where datediff(visited\_on,(select min(visited\_on) from Customer))>=6

select ad\_id,ifnull(round(sum(action="Clicked")/sum(action<>"Ignored")\*100,2),0) as ctr

from Ads

group by ad\_id

order by ctr desc, ad\_id

select PRODUCT\_NAME, t.unit UNIT

from Products p

join (

select product\_id,sum(unit) as unit

from Orders

where order\_date between '2020-02-01' and '2020-02-29'

group by product\_id

having sum(unit)>=100) t

on p.product\_id=t.product\_id

**1336**

select ifnull(count(t.amount),0)

from visits v

left join transactions t

on v.user\_id=t.user\_id and v.visit\_date=t.transaction\_date

group by v.user\_id,v.visit\_date

select (select u.name

from Users u

join(

select user\_id,count(movie\_id) as num

from Movie\_Rating

group by user\_id

order by count(movie\_id) desc, user\_id) t1

on t1.user\_id=u.user\_id

order by num desc, u.name limit 1) as results

union

select (select m.title

from Movies m

join(

select movie\_id,avg(rating) as rating

from Movie\_Rating

where created\_at between '2020-02-01' and '2020-02-29'

group by movie\_id

order by avg(rating) desc,movie\_id) t2

on m.movie\_id=t2.movie\_id

order by rating desc, m.title limit 1) as results

select id,name

from Students

where department\_id not in (select id from Departments)

select activity

from friends

group by activity

having count(\*) <> (select count(id) from friends group by activity order by count(id) desc limit 1) and count(\*) <> (select count(id) from friends group by activity order by count(id)  limit 1)

select invoice\_id,t.customer\_name,price,contacts\_cnt,trusted\_contacts\_cnt

from Invoices i

join(

select c1.customer\_id, c1.customer\_name, count(c2.user\_id) as contacts\_cnt,

count(case when c2.contact\_name in (select customer\_name from customers) then 1 else null end) as trusted\_contacts\_cnt

from customers c1

left join contacts c2

on c1.customer\_id=c2.user\_id

group by c1.customer\_id) t

on i.user\_id=t.customer\_id

order by invoice\_id

OR sum(if(c2.contact\_name in (select customer\_name from customers),1,0)) as nmb

select unique\_id,e1.name

from Employees e1

left join EmployeeUNI e2

on e1.id=e2.id

select username,activity,startDate,endDate

from(

select \*, row\_number() over(partition by username order by startDate desc) as rnk,

count(\*) over(partition by username) as num

from useractivity) t

where rnk=2 or num=1

**1384**

select unique\_id,e1.name

from Employees e1

left join EmployeeUNI e2

on e1.id=e2.id

select t.product\_id,p.product\_name,report\_year,total\_amount

from(

select product\_id,"2018" as report\_year,(datediff(if(period\_end<'2019-01-01',period\_end,date('2018-12-31')),if(period\_start>'2018-01-01',period\_start,date('2018-01-01')))+1)\*average\_daily\_sales as total\_amount

from sales

group by product\_id

union all

select product\_id,"2019" as report\_year,(datediff(if(period\_end<'2020-01-01',period\_end,date('2019-12-31')),if(period\_start>'2019-01-01',period\_start,date('2019-01-01')))+1)\*average\_daily\_sales as total\_amount

from sales

group by product\_id

union all

select product\_id,"2020" as report\_year,(datediff(if(period\_end<'2021-01-01',period\_end,date('2020-12-31')),if(period\_start>'2020-01-01',period\_start,date('2020-01-01')))+1)\*average\_daily\_sales as total\_amount

from sales

group by product\_id) t

join product p on t.product\_id=p.product\_id

where total\_amount>0

order by product\_id,report\_year

select stock\_name,

sum(case when operation="Buy" then -price else price end) as capital\_gain\_loss

from Stocks

group by stock\_name

**1398**

select o.customer\_id,c.customer\_name

from Orders o

join customers c

on o.customer\_id=c.customer\_id

group by customer\_id

having sum(product\_name="A")>0 and sum(product\_name="B")>0 and sum(product\_name="C")=0

select u.name, ifnull(sum(distance),0) as travelled\_distance

from users u

left join rides r

on u.id=r.user\_id

group by u.name

order by sum(distance) desc, u.name

select \*

from student

where student\_id not in

(select distinct student\_id

from

(select \*,dense\_rank() over(partition by exam\_id order by score) as rnk1,

dense\_rank() over(partition by exam\_id order by score desc) rnk2

from exam) t1

where rnk1=1 or rnk2=1) and student\_id in (select distinct student\_id from exam)

select q.id,q.year,ifnull(npv,0) as npv

from Queries q

left join NPV n

on q.id=n.id and q.year=n.year

select '[0-5>' as 'bin', count(\*) as total

from Sessions

where duration/60 < 5

union all

select '[5-10>' as 'bin', count(\*) as total

from Sessions

where duration/60 >=5 and duration/60 < 10

union all

select '[10-15>' as 'bin', count(\*) as total

from Sessions

where duration/60 >=10 and duration/60 < 15

union all

select '15 or more' as 'bin', count(\*) as total

from Sessions

where duration/60 >=15

select left\_operand,operator,right\_operand,

case when operator = ">" and v1.value>v2.value then "true"

    when operator="=" and v1.value=v2.value then "true"

    when operator="<" and v1.value<v2.value then "true"

    else "false"

    end as value

from Expressions e

left join Variables v1

on e.left\_operand=v1.name

left join Variables v2

on e.right\_operand=v2.name

select sale\_date, sum(case when fruit="apples" then sold\_num else null end)-sum(case when fruit="oranges" then sold\_num else null end) as diff

from sales

group by sale\_date

**1454**

select distinct a.id,a.name

from accounts a

join(

select id

from(

select id,login\_date,rnk,date\_sub(login\_date,interval rnk day) as rnk\_day

from(

select distinct id,login\_date,dense\_rank() over (partition by id order by login\_date) as rnk

from logins)  t) t1

group by id, rnk\_day

having count(\*)>=5) t2 on a.id=t2.id

order by id

select company\_id,employee\_id,employee\_name,

round(case when max(salary) over(partition by company\_id) < 1000 then salary

   when max(salary) over(partition by company\_id) between 1000 and 10000 then salary\*(1-0.24)

   else salary\*(1-0.49)

   end,0) as salary

from Salaries

select sell\_date, count(distinct product) as num\_sold, **group\_concat**(distinct product) as products

from Activities

group by sell\_date

select distinct title

from content c

left join TVProgram t

on c.content\_id=t.content\_id

where Kids\_content="Y" and content\_type="Movies" and program\_date between '2020-06-01' and '2020-06-30'

select c1.name country

from(

select caller\_id,duration from Calls

union all

select callee\_id,duration from Calls) c

join Person p on p.id=c.caller\_id

join country c1 on left(p.phone\_number,3)=c1.country\_code

group by c1.name

having avg(duration) > (select avg(duration) from calls)

select c.customer\_id,c.name

from customers c

join(

select customer\_id, ifnull(sum(case when order\_date between '2020-06-01' and '2020-06-30' then quantity\*price end),0) as june\_sum,ifnull(sum(case when order\_date between '2020-07-01' and '2020-07-31' then quantity\*price end),0) as july\_sum

from orders o

join product p on o.product\_id=p.product\_id

group by customer\_id) t

on t.customer\_id=c.customer\_id

where june\_sum>=100 and july\_sum>=100

**1517**

select \* from Users

where mail regexp '^[a-zA-Z]+[a-zA-Z0-9\_\\./\\-]{0,}@leetcode.com$'

order by user\_id

select \*

from Patients

where conditions like '%DIAB1%'

select c.name as customer\_name,c.customer\_id,t.order\_id,t.order\_date

from

(select customer\_id,order\_id,order\_date,row\_number() over(partition by customer\_id order by order\_date desc) as od from Orders) t

join customers c

on c.customer\_id=t.customer\_id

where od <=3

order by name,c.customer\_id,t.order\_date desc

select c.name as customer\_name,c.customer\_id,t.order\_id,t.order\_date

from

(select customer\_id,order\_id,order\_date,row\_number() over(partition by customer\_id order by order\_date desc) as od from Orders) t

join customers c

on c.customer\_id=t.customer\_id

where od <=3

order by name,c.customer\_id,t.order\_date desc

select trim(lower(product\_name)) as product\_name, date\_format(sale\_date,"%Y-%m") as sale\_date,

count(sale\_id) as total

from sales

group by trim(lower(product\_name)),date\_format(sale\_date,"%Y-%m")

order by product\_name,sale\_date

select p.product\_name,p.product\_id,t.order\_id,t.order\_date

from Products p

join(

select order\_id,product\_id,order\_date,dense\_rank() over(partition by product\_id order by order\_date desc) as rk

from Orders) t

on p.product\_id=t.product\_id

where rk=1

order by p.product\_name,p.product\_id,t.order\_id

select user\_id,user\_name, credit+ifnull(amount,0) as credit, if(credit+ifnull(amount,0)>0,"No","Yes") as credit\_limit\_breached

from users u

left join(

select paid\_by as id, sum(t.amount) as amount

from(

select paid\_by,paid\_to,-amount as amount from Transactions

union all

select paid\_to,paid\_by,amount as amount from Transactions

) t

group by paid\_by) t1

on t1.id=u.user\_id

select date\_format(order\_date,"%Y-%m") as month,

    count(distinct order\_id) as order\_count,

    count(distinct customer\_id) customer\_count

from Orders

where invoice>20

group by date\_format(order\_date,"%Y-%m")

select w.name as warehouse\_name, sum(volume\*units) as volume

from warehouse w

left join(

select \*,width\*length\*height as volume from products) t on t.product\_id=w.product\_id

group by w.name

select customer\_id,count(visit\_id) as count\_no\_trans

from visits

where visit\_id not in (select visit\_id from Transactions)

group by customer\_id